AD-A126 198 THREE PAPERS ON RULE-BASED ESTIMATION OF STATISTICS ON DATABASES(U) STANFORD UNIV CA DEPT OF COMPUTER SCIENCE N C ROWE OCT 82 STAN-CS-82-948 N00039-82-G-0250 1/1

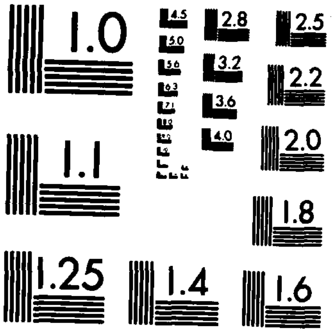
UNCLASSIFIED F/G 5/2 NL

END

FILMED

[illegible]

DTIC



MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS-1963-A

October 1982

Report No. STAN-CS-82-948

3

Three Papers on Rule-Based Estimation of Statistics on Databases

by

Neil C. Rowe

Department of Computer Science

Stanford University
Stanford, CA 94305

Three papers on rule-based estimation of statistics on databases

Neil C. Rowe
Department of Computer Science
Stanford University
Stanford, CA 94305

This work is part of the Knowledge Base Management Systems Project, under contract # N00039-82-G-0250 from the Defense Advanced Research Projects Agency of the United States Department of Defense. The views and conclusions contained in this document are those of the author and should not be interpreted as representative of the official policies of DARPA or the US Government.

Table of Contents

Introduction 1

1. Rule-based statistical calculations on a "database abstract" 2

1.1 Introduction 3
1.1.1 Precomputed statistics 3
1.1.2 Exploiting redundancy 3
1.1.3 A database abstract 4
1.1.4 Outline of the rest of the paper 4
1.2 A rule-based system for inferring statistics 4
1.2.1 Assumptions 4
1.2.2 Some exact rules 5
1.2.3 Two ways to interpret rules 5
1.2.4 Probability-distribution rules 6
1.2.5 Sampling-theory probability-distribution rules 7
1.2.6 Combining probability distributions 7
1.2.7 Rules for derived fields 8
1.2.8 Rules for nulls 8
1.2.9 Multi-field rules 8
1.2.10 Rules for upwards and lateral inheritance 9
1.2.11 How the rules are used 9
1.3 Building the database abstract 9
1.3.1 Psychological analogies 10
1.3.2 General approach to building the database abstract 10
1.3.3 Evaluating databases for this approach 10
1.4 Updating the database abstract 11
1.4.1 Updating explicit statistics 11
1.4.2 Updating implicit statistics 11
1.5 Supporting high-level queries 12
1.5.1 Fuzzy queries and answers 12
1.5.2 Queries on prototypes 12
1.5.3 Handling analogies 13
1.6 The database abstract in context 13
1.6.1 Distributed systems 13
1.6.2 The database abstract as extensional description 13
1.7 Conclusiorıs 13
1.8 Acknowledgments 14
1.9 Appendix: a partial implementation 14

2. Inheritance of statistical properties 19

2.1 Motivation 19
2.2 Our four-characteristic approach 20
2.3 Inheritance types 21
2.4 Closed-world inferences 22
2.5 A production system architecture 22
2.6 An application 23

3. Diophantine compromise of a statistical database 24

3.1 Problem definition 24
3.2 Solving the equations 25
3.3 Additional evidence: multiple equations 26
3.3.1 Equations from moments 26
3.3.2 Equations from linked attributes 26
3.3.3 Equations from one-way maps 27
3.3.4 Equations from nonnumeric attributes 28
3.3.5 Equations from virtual attributes 28
3.3.6 Equations from database updates 28
3.3.7 Equations from multiple factorizations 29
3.4 Additional evidence: multiple constraints 30
3.5 Countermeasures 31
3.6 Other uses of Diophantine inference 32
3.7 Conclusions 32

Introduction

This report contains three papers on rule-based estimation of statistics on a database: an overview, followed by two more specialized papers. The first, "Rule-based Statistical Calculations on a Database Abstract", is addressed to a general database audience and was presented at the First LBL Workshop on Statistical Database Management, Menlo Park, California, December 1981. The second, "Inheritance of Statistical Properties", is addressed to an artificial intelligence audience and was presented at the National Conference of the American Association for Artificial Intelligence, Pittsburgh, Pennsylvania, August 1982. The third, "Diophantine Compromise of a Statistical Database", is addressed to an audience of database theorists.

1. Rule-based statistical calculations on a "database abstract"

Statistical calculations can take large amounts of time in a large database. Precomputed summary values may be stored to improve access speed, but they can only cover a small fraction of the great number of sets queriable. We suggest a compromise using some precomputed statistics together with a set of inference rules that estimate other, unrepresented statistics in terms of the precomputed ones. Such a configuration can be constructed so as to guarantee a certain level of accuracy for a large class of calculations of simple descriptive statistics on the database. We discuss the architecture of such a system, the kinds of rules it must contain, and how it can be constructed and updated. We then comment on some broader implications.

1.1 Introduction

There are many applications of simple aggregate statistics -- averages, dispersions, set sizes, etc. -- on very large databases. But for many such databases explicit calculations require enormous amounts of time. Rarely is there a principal dimension of the database for which most queries are asked, and thus large access delays to essentially random places on secondary storage occur in obtaining each relevant record. Since good statistical results depend on large quantities of data, the time to execute most simple aggregate-statistical queries for a large general database is minutes or hours, several orders of magnitude more than typical single-record nonstatistical querying.

1.1.1 Precomputed statistics

A way to avoid large access delays is to precompute certain statistics that a user might be likely to ask about (i.e., store a kind of "derived data" [24]). Then lookup replaces calculating. But if we have m independent subsets of a database that users can refer to by a simple name ("first-order" sets), then there are $O(m^2)$ "second-order" sets consisting of the intersections of those sets. Similarly, there are $O(m^3)$ "third-order" sets representing the intersection of any three simple sets, and so on.

This exponential explosion of set intersections is critical because many of the interesting statistical queries that users generally like to ask will be about complex intersections, unions, and complements of several sets. This reflects a frequent user motivation of attempting to isolate central causes and effects in data by holding a number of factors constant while varying others. Thus, it seems clear that in large databases, very few of all possible statistical queries can be answerable from previously derived results even if large amounts of storage are set aside for such results -- there are just too many. (And we have not even considered the very large number of first-order sets that can be created by partitioning a continuous distribution of values.)

1.1.2 Exploiting redundancy

Hence we will be unlikely to be able to keep sufficient quantities of precomputed statistics around to make much of a dent in the bulk of those needed. However, there may be a way out: some statistics are partially redundant. For instance, we expect the set of American tankers to have similarities to both the set of American ships and the set of tankers. Perhaps we could average the average sizes of American ships and tankers to get a good estimate of the average size of American tankers. Then we might not need storage for the latter statistic at all.

1.1.3 A database abstract

We envision, then, a collection of broadly useful precomputed statistics, a "database abstract", together with a set of rules. Like literary "abstracts", the database abstract will try to represent both what is particularly typical and what is particularly unusual in the data. When a user asks a statistical query, we first try to answer it with precomputed data in the abstract. If the needed data isn't there, we then try to derive it using rules. If we cannot get an exact value, we try to get as narrow bounds on the value as possible, using all applicable rules. We then return this value to the user.

1.1.4 Outline of the rest of the paper

We first discuss what our statistical inference rules look like and how they are applied. Next, we explain how the database abstract is initially loaded, and how it is dynamically updated. We comment on how simple descriptive statistics can be used to answer higher-level queries. Finally, we give an overview of the entire approach.

1.2 A rule-based system for inferring statistics

Our approach is to develop a set a rules for deriving values of one statistic on one set from values of other statistics on other sets. Such an approach is in the tradition of rule-based systems such as MYCIN [19] and PROSPECTOR [9].

We eventually expect to have several hundred such rules. The Appendix shows the current status of a partial implementation.

1.2.1 Assumptions

We assume here a query capability equivalent to the propositional calculus plus the simple statistical (aggregate) operators. That is, queries consist of specifications of a set of database tuples and fields to which an aggregate statistical operator is applied. The target set is specified without quantifiers, just with AND (set intersection), OR (set union), and NOT (set complement). We assume statistical operators like SIZE (set size), MEAN, MEDIAN, MODE, MAX, MIN, SIGMA (standard deviation), MODEFREQUENCY, etc.

Any query involving the above logical and statistical operators can be treated as a set of single-statistical-operator subproblems by appropriate decomposition. For instance, the query asking for the number of ships longer than the average Ameiican tanker can be decomposed into a query of the MEAN of the lengths of American tankers, followed by a query of the SIZE of the set of ships longer than this now-known number.

We further assume a database consisting of a single relation. (Many databases can be seen as such in a "universal relation" form.)

1.2.2 Some exact rules

Let A, B, C, etc. be sets of database tuples. Using a prefix notation for queries, we can express (OR A B) as the set of all tuples that are in either A or B; (MAX A F) as the largest value for the field F of the set A; and so on. So some simple rules are:

```
(MAX (OR A B) F) = (LARGEROF (MAX A F) (MAX B F))                      (1.1)
(MIN (OR A B) F) = (SMALLEROF (MIN A F) (MIN B F))                     (1.2)
```

where LARGEROF and SMALLEROF represent the larger and smaller of a pair of numbers (not sets)

```
(SIZE (OR A B)) = (DIFFERENCE (PLUS (SIZE A) (SIZE B)) (SIZE (AND A B)))   (1.3)
```

$[S_A + S_B - S_{A\,B}]$

```
(MEAN (OR A B) F) = (QUOTIENT (DIFFERENCE                                  (1.4)
                    (PLUS (TIMES (MEAN A F) (SIZE A)) (TIMES (MEAN B F) (SIZE B)))
                    (TIMES (MEAN (AND A B) F) (SIZE (AND A B))) )
     (SIZE (OR A B)))
```

$[(\mu_{AF}S_A + \mu_{BF}S_B - \mu_{(A\,B)F}S_{(A\,B)}) / S_{AUB}]$

```
(SIZE (NOT A)) = (DIFFERENCE UNIVERSESIZE (SIZE A))                        (1.5)
(MEAN (NOT A) F) = (QUOTIENT (DIFFERENCE                                   (1.6)
                    (TIMES UNIVERSEMEAN UNIVERSESIZE)
                    (TIMES (MEAN A F) (SIZE A)))
     (SIZE (NOT A)) )
```

where UNIVERSESIZE is the total number of tuples in the database, and UNIVERSEMEAN is their mean.

1.2.3 Two ways to interpret rules

There are two ways to interpret the preceding. In one sense we are doing a new form of database query "optimization" (cf. [23]). We are rearranging the order of statistical and logical operations in a query; specifically, trying to "push inside as far as possible" statistical operators. For instance in rule ((1.1)), we are replacing a MAX on a union by a simple arithmetic operator applied to two MAXs on two simpler sets. The idea, then, is to get the statistical operators applied to simple sets as much as possible, not the many complicated sets described by ORs, ANDs, and NOTs, which are less likely to have precomputed statistics. However, there is an important difference here from classical optimization: once we have made the decision to compute on the database abstract and not the full database, we are not concerned with optimizing access time any further, only *accuracy* (which, as we shall see subsequently, may be different for different "inexact" rules used). The number of records that need be retrieved from the database abstract in processing a query will be relatively small, and access times will not vary much.

Another way to look at this is as a kind of "property inheritance" as the term is used in artificial intelligence. We don't want to store statistical properties of every conceivable set. Rather, we want to "inherit" properties from more general "parent" sets where possible, using rules to accommodate evidence from multiple parents, etc. (AND A B) is a sibling of the sets A and B in this sense. [17] illustrates the complexity of the inheritance methods we need here, however, because rarely is the exact value for a statistic inferable from "parents".

1.2.4 Probability-distribution rules

There aren't very many exact rules of the (2.2) type. But in many cases users do not need or want an exact answer, just one with some certain amount of accuracy. A frequent use of database statistics is to monitor ongoing activities and situations for problems or abnormalities. In this case large amounts of data need to be scanned, and quick but approximate answers may help identify the few problem areas for later detailed study with the full database.

Approximate answers are also characteristic of the way people answer questions. Fuzzy terms like "big", "small", "common", "rare", "average", etc. are natural to use. Thus an approximate answer as conveyed by a fuzzy English description can be a highly "cooperative" way of answering a question.

Our basic approach is to give an exact answer to a query if possible, otherwise a quadruple characterizing the probability distribution of the answer. This quadruple contains, in order, the maximum, minimum, estimate, and expected standard deviation of the estimate when the statistic in question is numeric; and the mode, mode frequency, number of distinct values, and general symbol describing all values, when the statistic in question is non-numeric.

In what follows we shall discuss almost exclusively numeric statistics. We shall use the notation

- (SUP-S A) = the largest possible value the statistic S of A with respect to field F could take
- (INF- A) = the smallest possible value the statistic S of A with respect to F could take
- (EST-S A) = a best estimate of the value of statistic S of A with respect to field F
- (ERR-S A) = the standard deviation associated with (EST-S A)

1.2.5 Sampling-theory probability-distribution rules

An important class of inexact rules have to do with subsets of larger sets. For instance:

(SUP-MAX A F) = (MAX B F) if A is a subset of B (1.7)

(MIN-MAX A F) = (MIN B F) if A is a subset of B

These rules have the important corollaries:

(SUP-MAX (AND A B) F) = (SMALLEROF (MAX A F) (MAX B F)) (1.8)

(MIN-MAX (AND A B) F) = (LARGEROF (MIN A F) (MIN B F)) (1.9)

which show analogies to our rules for OR, ((1.1)) and ((1.2)).

Similarly for the SIZE (set size) statistic,

(SUP-SIZE A) = (SIZE B) if A is a subset of B (1.10)

(INF-IZE A) = 0 (1.11)

hence:

(SUP-SIZE (AND A B)) = (SMALLEROF (SIZE A) (SIZE B)) (1.12)

(INF-IZE (AND A B)) = 0 (1.13)

Also note from sampling theory:

(EST-SIZE (AND A B)) = (QUOTIENT (TIMES (SIZE A) (SIZE B)) UNIVERSESIZE) (1.14)
assuming that memberships in A and B are statistically independent

(EST-MEAN A F) = (MEAN B F) if A is a statistically random sample from B (1.15)

(ERR-MEAN A F) = (TIMES (SIGMA B F) (SQRT (QUOTIENT (1.16)
(DIFFERENCE (SIZE B) (SIZE A))
(TIMES (SUB1 (SIZE B)) (SIZE A)))))
$[\sigma_{BF}((S_B - S_A)/(S_B \cdot 1)S_A)^{1/2}]$

1.2.6 Combining probability distributions

Probability distributions can be combined in the usual ways from probability theory. A function like PLUS can be generalized to do convolution when both its components are distributions, for instance. So (PLUS A B) where A has distribution (100 0 40 10) and B has distribution (50 0 10 10) is (150 0 50 14.7), using our (MAX MIN MEAN SIGMA) notation and assuming close-to-normal distributions about the means.

As a corollary this allows us to handle databases with inexact data entries. Recorded values can have an associated degree of accuracy that can be expressed either as a range or a mean and standard deviation. This inaccuracy can be combined algebraically with that of the rules, and handled with no further amount of effort.

1.2.7 Rules for derived fields

Fields that can be derived from others (e.g., a field that is the sum of two others) can be handled by the same mechanism, though we do not have the space to illustrate this rich topic here. In fact, a related idea can be used to relate two explicitly represented database fields that are only correlated, as dimensions of a product in an inventory database with weight. Such numeric correlations can be characterized by a regression with an associated formula, standard error of estimate, and absolute errors, and this information can be expressed in a quadruple similar to that [illegible]d for describing distributions.

Something analogous can be done for non-numeric fields. Given hierarch[illegible] abstraction on values of fields, we can create a set of "generalized tuples" expressing the perm[illegible]ble combinations of categories of values on different fields.

1.2.8 Rules for nulls

Another important class of rules provides a way to handle "value-unknown" null values in the actual database. Define a function (NULLREMOVE A F) which gives all the members of a set A that do not have null for their value of field F, and its "twin" function (NULLITEMS A F) that gives all other members of A. We now put the additional restriction on all the previous rules that they only apply to sets without nulls in the field in question, and supply the following new rules:

```
(MIN-MAX A F) = (MAX (NULLREMOVE A) F)                                     (1.17)
(SUP-MAX A F) = (ABSMAX (NULLREMOVE A) F)                                  (1.18)
where ABSMAX denotes the "largest conceivable" value
the set could have, in the absence of all a priori information.
(SUP-MEAN A F) = (QUOTIENT                                                 (1.19)
(PLUS              (TIMES (SIZE (NULLREMOVE A)) (MEAN (NULLREMOVE A) F))
          (TIMES (SIZE (NULLITEMS A)) (ABSMAX A F)) )
     (SIZE A) )
```

1.2.9 Multi-field rules

There are also rules involving more than one field, such as:

```
(SUP-SIZE A) = minimum over the set of all fields F of                     (1.20)
(SIZE (RANGE-FIELD (MIN A F) (MAX A F) F))
```

where RANGE-FIELD denotes the set covering the range of its first two arguments on the field denoted by the third. So if F is ship length, (RANGE-FIELD 300 500 LENGTH) denotes the set of all ships between 300 and 500 in length.

1.2.10 Rules for upwards and lateral inheritance

The rules mentioned so far are for "downwards" inheritance of statistics from more general sets to less general sets. In some cases other kinds of inheritance may apply too. "Upwards" inheritance occurs when we know statistics for the complete set of subsets of a given set:

A = (OR B C) if every member of A is also a member of B or C (1.21)

Hence:

```
(MEAN A F) = (QUOTIENT
        (PLUS (TIMES (MEAN B F) (SIZE B)) (TIMES (MEAN C F) (SIZE C)))
        (SIZE A))
and
(MAX A F) = (LARGEROF (MAX B F) (MAX C F))
if every member of A is also a member of B or C
```

"Lateral inheritance" ([2]) is inheritance between subsets of some set. It is useful when several subsets are very similar except for a few key features. In this case normal downwards inheritance may not work too well if those features are atypical of the more general parent set. To implement this, predefine a set of domain-dependent rules explaining how set A is just like set B except for the x feature. An example would be defining supertankers as just like tankers only proportionately larger.

1.2.11 How the rules are used

When a user issues a query we either fetch the answer directly if it has been stored, or use our rules to derive it from other statistics. Several rules may apply to the same query, and the application of one rule may make it possible to apply many more. In general, then, we have a "production system" in the sense of [7]. We attempt to explore many paths in parallel using a "breadth-first search" strategy, combining the results into a single cumulative estimate of the statistic in question.

1.3 Building the database abstract

In constructing the database abstract we would like to guarantee a certain level of accuracy in statistical estimates -- if not for all queries, at least for those up to some maximum complexity. Granted, we can't always look up directly the statistic the user wants or get it from exact rules, we still would like to ensure we get, say for a numeric statistic, within 10% of the value from explicit calculation in the full database. We can ensure this by building the database abstract carefully. We shall store statistics for all first-order sets, plus statistics for other sets that are "atypical", i.e. cannot be very well derived by rules from other sets. For instance, the mean American tanker length is atypical if it is much lower than the means for tankers and American ships. These sets can be arbitrary combinations of ANDs, ORs, and NOTs.

1.3.1 Psychological analogies

Human knowledge concerning large bodies of facts seems to be structured on somewhat analogous principles. People store reasonably detailed information about large general categories of things, but only store information for more specific categories that is in some sense unusual. In particular, Collins ([6, 5]) cites extensive use of subpart inheritance analogous to ((1.15)) and closed-world assumptions similar to ((1.21)) in human reasoning. We are using many of his ideas in our work.

1.3.2 General approach to building the database abstract

Generally speaking, we attempt to linearize the loading problem into an examination of each set separately. For s statistics, there are 2^s options for representing each set (either including or excluding each statistic). We attempt to choose an option that captures the essence of the set without too many details. It should be an option that allows 10% accuracy on all s statistics, as verified by comparision with an actual calculation on the database. To make this choice we are exploring partial orderings of statistics, linear decomposition, semantic heuristics, and a variety of other ideas. This is an active area of current research and we do not claim to have a complete solution yet.

Note that loading need only be done once, when the database is originally set up. Updating the database abstract over time is discussed in section 1.4. Note also that the number of possible subsets of a database is unbounded since there are arbitrarily complex intersections and complements of sets. (The same subset may have many different descriptions, but there is no general way to recognize them without fetching the tuples themselves.) So a size limit is needed -- say 10 items -- below which a set will not be considered for tabulation of any statistics. Such sets are so small as to have unhelpful statistics anyway.

1.3.3 Evaluating databases for this approach

Loading the database abstract will be computationally the most difficult part of our approach. Clearly not all databases are tractable in this regard. Unless we can derive most statistics by rules from other statistics, the database abstract may be the same order of magnitude as full database and consequently difficult to store. Generally speaking, it is databases where there are strongly correlated classes that are the problem. Though we do have rules for handling simple field-to-field linear correlations (see 1.2.7), databases with many strong and complex correlations between fields will be difficult to handle.

1.4 Updating the database abstract

Database values may change with time, and the database abstract must be updated in turn to reflect those changes. (Though generally speaking data that is updated frequently is inappropriate to represent in a database abstract, since required frequent recomputation negates the benefits of precomputation.)

There are two basic kinds of update, to statistics explicitly and implicitly represented.

1.4.1 Updating explicit statistics

When a statistic is explicitly represented in the database abstract, update can frequently be done dynamically. Here are some examples:

- Set size: just add or subtract 1 from associated database abstract sets' size statistic on database insert or delete.
- Mean: adjust old means by weighting in inserts, weighting out deletes.
- Maximum and minimum: on insert, change old value if new value exceeds bounds. On delete, go to database to recompute if deleted value is the old max or min.
- Mode: compare two differences: between frequency of two most common items, and between the maximum net number of inserts for an item and the net number of inserts for the mode. Go to database to recompute when latter exceeds former.

1.4.2 Updating implicit statistics

When a statistic is not explicitly kept, but is expected to be derived by rules from other statistics, updating is a bit different. Given a 10% accuracy criterion that all derived statistics on sets more than ten items are guaranteed to fulfill, we cannot be sure that an update will keep a statistical value in this 10% region, because the original value may have been close to the edge. But if we use a weaker criterion once updates have occurred, like 20% accuracy, we can still offer the user *some* guarantee without having to recheck the database on every update.

So for set size, for instance, we keep track of the difference between the number of adds and deletes to a set. If the absolute value of this exceeds 5% of the estimated set size as calculated from rules, only then do we go to the database to recompute the value, and adjust the database abstract. For the mean similarly, we track the average of all the adds minus the average of all the deletes. If this, weighted appropriately, asserts more than a 5% change on the original rule-computed mean, go to the database to recalculate. But for maximum and minimum we never have to go to the database: we just change the database abstract value if an update value is 10% beyond the existing bounds.

We must also track changes in parent sets affecting siblings. Since many of these have linear effects on siblings, most of the time we can follow a 5% recompute criterion for parent sets too. Thus with updates one can guarantee a continuous 20% accuracy.

1.5 Supporting high-level queries

A database abstract can support other kinds of queries besides the obvious ones about ranges, averages, and dispersions. In particular, it can also support more informal, "browsing" usage.

1.5.1 Fuzzy queries and answers

Since uncertainty in statistical values is fully addressed in this system, "fuzzy" notions such as "big", "most", and "frequently" can be incorporated by either definitions of subjective probability distributions or the "possibility" approach of fuzzy-set theory ([25]). The query "How small are the American tankers?" could be answered, "They're quite small." So fuzzy terms can both be understood on input and used to make output more natural.

1.5.2 Queries on prototypes

The database abstract information about a set is close to what is needed to construct a "prototype" for a set. While there are many complicated and controversial issues regarding, say, whether the prototypical tanker has the mean, median, or mode length, or whether prototypes must be actual database tuples, we can construct a first approximation of a prototype as an entity with the mean value for numeric fields and the mode value for non-numeric fields.

Prototypes have had some attention in artificial intelligence. However, our approach brings in a new idea previously little recognized, that of the importance of knowledge of the dispersion of a field. Consider the set of tankers. If the average tanker is longer than the aveiage ship, that may or may not be significant. But if we know the standard deviation of ship lengths, we can see the degree of significance in terms of how many standard deviations average tanker length is from average ship length. So dispersion statistics of a set quantify what properties of the set's members are "significant", helpful information in evaluating prototypes. Note that exact statistical values are not necessary to perform this analysis, only reasonably good ones.

1.5.3 Handling analogies

The above suggests the more general capability of handling queries involving analogies. Given two sets, we can say which properties are significantly different, using the mean, standard deviation, and set size information for instance for numeric fields. We can handle both analogies explicit in a query (e.g. "How are supertankers different from regular tankers?") and clarifying analogies in the output (e.g. "Container ships have pretty much the same range of nationalities as those freighters you just looked at.")

1.6 The database abstract in context

We now take a broader view of the system of database abstract plus rules.

1.6.1 Distributed systems

One appealing aspect of putting statistics into a data structure distinct from the database is that it can be physically removed to a remote site as well. Since statistics are often the only concern of a large class of users, these users can be given their own machine and much faster response times, with updating by one-way data transmission from the central database. Security is enhanced by this arrangement, since the database abstract will only keep statistics on sufficiently different sets larger than some minimum size, making attempts to compromise protected information, while still occasionally possible, much more difficult. Thus it may be easier for unrestricted classes of users, like the general public, to be given access.

1.6.2 The database abstract as extensional description

The database abstract can also be viewed as a kind of database schema. Unlike the conceptual schema ([20]), it represents an abstraction of the database *contents*, not structural constraints on those contents. This is the distinction of "extensional" vs. "intensional" knowledge. Since database contents reflect logical limitations of database world, extensional information supports intensional. In fact, analysis of the database abstract may be able to suggest previously unrecognized intensional constraints on the database.

1.7 Conclusions

Work in artificial intelligence suggests that large amounts of knowledge must be richly structured for effective and flexible information retrieval. Simple descriptive statistics, as one kind of knowledge, have not generally been structured in information systems, in part due to the ease at which they can be derived from existing data. But in large databases the cost of such derivation may be very large, and more intelligent ways of getting the information must be found.

What we have presented here is one possible way, a mixture of precomputed statistics and inference rules. The rules greatly extend the coverage of the precomputed statistics, allowing one to represent only what is particularly noteworthy in a set of highly redundant data. This approach allows users to trade off some accuracy in statistical querying for greatly improved answer speed. It is hoped this research will allow greater use of large statistical databases.

1.8 Acknowledgments

Jerry Kaplan and Gio Wiederhold helped develop these ideas.

1.9 Appendix: a partial implementation

The following protocol demonstrates a few simple features of an INTERLISP implementation under development. Statistics have been derived in advance from a merchant shipping database. The database abstract consists only of information on the first-order classes "US" (United States registry ships), "USSR" (Russian registry), "INMED" (ships in the Mediterranean), "SEA", and "MIS" (specific subclasses of tankers). Thirty or so rules are used in answering the following queries on the database abstract.

In general, (ANS <set> <statistic> <field>) queries the <statistic> on the values of <set> with respect to <field>. ("SIZE" is set size, which does not require a <field>.) AND is set intersection, OR set union, and NOT set complement.

```
(ANS 'US 'SIZE)
15

(ANS 'USSR 'SIZE)
149

(ANS '(OR US USSR) 'SIZE)
164

(ANS '(AND US USSR) 'SIZE]
0

(ANS 'UNIVERSE 'SIZE)
605

(ANS '(NOT US) 'SIZE)
590

(ANS 'SEA 'SIZE)
121

(ANS '(AND USSR SEA) 'SIZE)
30
```

[Estimates are given as (upperlimit lowerlimit estimate estimatesigma). NAT is ship nationality, LAT latitude, LONG longitude.]

```
(ANS '(AND US SEA) 'SIZE)
(8 1 3.0 1.631812)

(ANS 'SEA 'MODE 'NAT)
USSR

(ANS 'SEA 'MODEFREQ 'NAT)
30

(ANS 'USSR 'MEAN 'LONG)
36.06711

(ANS 'USSR 'MEAN 'LAT)
36.04698

(ANS 'SEA 'MEAN 'LAT)
30.49587

(ANS '(AND USSR SEA) 'MEAN 'LAT)
(54 0 33.27143 16.01041)

(ANS '(OR USSR SEA) 'MEAN 'LAT)
(37.75417 31.00417 33.59524 2.001301)

(ANS '(AND USSR SEA) 'SIGMA 'LAT)
(28.06461 9.859006 15.70503 1.788096)
```

```
(ANS 'USSR 'MAX 'LAT)
64

(ANS 'USSR 'MIN 'LAT)
-60

(ANS 'SEA 'MAX 'LAT)
68

(ANS 'SEA 'MIN 'LAT)
-71

(ANS '(AND USSR SEA) 'MAX 'LAT)
(64 0 62.36275 1.637249)

(ANS '(AND USSR SEA) 'MIN 'LAT)
0

(ANS '(OR USSR SEA) 'MAX 'LAT)
68

(ANS 'UNIVERSE 'MAX 'LAT)
60

(ANS '(NOT USSR) 'MAX 'LAT)
60
```

```
(ANS 'USSR 'MEDIAN 'LAT)
37

(ANS 'SEA 'MEDIAN 'LAT)
33

(ANS 'MIS 'MEDIAN 'LAT)
33

(ANS '(OR USSR SEA) 'MEDIAN 'LAT)
(60 31.00417 36.93338 7.019957)

(ANS '(OR SEA MIS) 'MEDIAN 'LAT)
33

(ANS 'INMED 'SIZE)
366

(ANS '(AND SEA (AND USSR INMED)) 'SIZE)
(69 0 7.0 2.469316)

(ANS '(OR (AND SEA USSR) INMED) 'MEAN 'LAT)
(34.25956 24.61868 30.02373 2.757437)
```

2. Inheritance of statistical properties

Statistical aggregate properties (e.g. mean, maximum, mode) have not previously been thought to "inherit" between sets. But they do in a weak sense, and a collection of such "weak" information can be combined in a rule-based architecture to get stronger information.

2.1 Motivation

Suppose we have conducted a census of all elephants in the world and we can definitely say that all elephants are gray. Then by set-to-subset inheritance of the "color" property, the set of elephants in Clyde's herd must be gray, Clyde's herd being some particular herd of elephants.

This will not work for statistical aggregate properties such as maximum and mean. Suppose our census found that the longest elephant in the world is 27 feet long, and the average elephant 15 feet. This does *not* mean the longest elephant in Clyde's herd is 27 feet, nor the average in the herd 15 feet. But a weak form of inheritance is present, for we can assign different degrees of likelihood to the following:

1. "The longest elephant in Clyde's herd is 30 feet long."
2. "The average elephant in Clyde's herd is 30 feet long."
3. "The longest elephant in Clyde's herd is 27 feet long."
4. "The average elephant in Clyde's herd is 27 feet long."
5. "The longest elephant in Clyde's herd is 16 feet long."
6. "The average elephant in Clyde's herd is 16 feet long."

Statements 1 and 2 are impossible. Statement 3 is possible but a bit unlikely, whereas statement 4 is almost certainly impossible. Statement 5 is surprising and hence apparently unlikely, whereas 6 is quite reasonable. Since we don't know anything of Clyde's herd other than that they are elephants, a kind of inheritance from the properties of elephants in general must be happening.

The issue here is more important than elephants. Thousands of databases in existence support statistical questions about their contents. Exact answers to such questions may be very time-consuming for large data sets and/or remote access. Many users, especially non- statisticians, may be willing instead to accept much faster approximate answers via inheritance methods [15].

2.2 Our four-characteristic approach

We wish to address inheritance of the set properties maximum, mean, standard deviation, median, mode, fits to simple distributions, and correlations between different values of the same item. Our theory concerns set representation only (but sets of cardinality one can represent individuals). It concerns "definitional" sets primarily (those with absolute criteria for membership) as opposed to "natural kind" sets [1] (though degrees of set membership as in fuzzy set theory could be introduced). The theory mainly deals with extensions (exemplars), not intensions (meanings). It also only addresses the set-subset semantic relationship; however, often other relationships can be viewed this way by "atomization" of the included concepts, e.g. geographical containment may be seen as a set-subset relationship between sets of points.

The key is to note that while in a few cases statistical properties inherit values exactly from set to set, in most cases they do not; but that there are characterizations of a numeric statistic that will inherit much more often:

- an upper bound on its value
- a lower bound on its value
- a best estimate of the value
- a standard deviation of possibilities for the value

Some examples:

- An upper bound on the mean of a subset is the maximum of the set.
- A lower bound on the maximum of a subset is the minimum of the set.
- A best estimate of the mean of a subset, in the absence of further information, is the mean of the set.
- A standard deviation of the mean of a subset is approximately the standard deviation of the set times the square root of the difference of the reciprocals of the subset size and set size.

The last also illustrates an important feature of statistical property inheritance, namely that functions (in the mathematical sense) of values may be inherited rather than the values themselves. But since the different values are so strongly coupled it seems fair to still call it "inheritance".

Inheritance of nonnumeric statistics such as mode can analogously be characterized by a best estimate, a superset guaranteed to include all values, and an estimated relative frequency of the estimate among all possible values. Note this approach is a distinct alternative to often-arguable certainty factors for specifying partial knowledge.

2.3 Inheritance types

There are three "dimensions" of statistical inheritance: what statistic it concerns, which of the four abovementioned manifestations it addresses, and how it basically works. The main categories of the latter are:

- Downwards inheritance. That is, from set to subset, as in the examples of the last section. This is the usual direction for statistical inheritance since it is usually the direction of greatest fanout: people tend to store information more for general concepts than specific concepts, for broadest utility. In particular, downwards inheritance from sets to their intersection is very common in human reasoning, much more so than reasoning with unions and complements of sets.

- Upwards inheritance. Inheritance from subset to set occurs with set unions, in particular unions of disjoint sets which (a) seem easier for humans to grasp, and (b) have many nice inheritance properties (e.g. the largest elephant is the larger of the largest male and largest female elephants). Sampling, random or otherwise, to estimate characteristics of a population is another form of upwards inheritance, though with the special disadvantage of involving a non-definitional set. Upwards inheritance also arises with caching [12]. People may cache data on some small subsets important to them (like Clyde's herd) in addition to general-purpose data. Upwards (as well as downwards) inheritance is helpful for dealing with "intermediate" concepts above the cache but below general-purpose knowledge (e.g. the set of elephants on Clyde's rangelands).

- Lateral inheritance. A set can suggest characteristics of sibling sets of the same parent superset [2]. Two examples are set complements (i.e. the set of all items not in a set, with respect to some universe), and when sibling sets differ only by an independent variable such as time or space, and there are constraints on the rate of change (i.e. derivatives) of numeric attributes between siblings (e.g. the stock market average on successive days).

- Diagonal inheritance. An interesting hybrid of downwards and lateral inheritance is possible with statistical properties. Given statistics on the parent and some set of siblings, we can often "subtract" out the effect of the known siblings from the parent to get better estimates on the unknown siblings. For instance, the number of female elephants is the total number of elephants minus the number of male elephants. This also works for moment and extrema statistics.

- Intra-concept inheritance. Inheritance can also occur between different statistics on the same set, if certain statistics are more "basic" than others. For instance, mean can be estimated as the average of maximum and minimum, and thus can be said to "inherit" from them; people may reason this way, as in guessing of the center of a visual object from its contours. But in principle almost any direction is possible with numerical and nonnumerical relaxation techniques.

- Value-description-level inheritance. Real-world property values, especially nonnumeric ones, can be grouped at different levels of detail, and inheritance is possible between levels for the same set and same statistic. For instance, the number of different herds can be estimated from the number of different elephants and general knowledge of how many elephants are in a herd.

- Inheritance-rule inheritance. Some sets are sufficiently "special" to have additional inheritance rules for all subsets or supersets. An example is an all-integer set, where for any subset an upper bound on the number of distinct values for that property is the ceiling on the range.

2.4 Closed-world inferences

Since there are many statistics, and even a small set can have many subsets, default reasoning is essential for efficiency with statistical properties. Inferences from the absence of explicit memory information are common in human reasoning [5]; in particular, the idea that "sufficiently important" sets whose statistics are not explicitly noted must be not "unusual" in regard to those statistics. We can define "sufficiently important" and "unusual" relative to what inheritance predicts.

2.5 A production system architecture

So many different kinds of inheritance (even just those applicable to the same concept), complicated combination and cascading of different inheritances, inheritance of functions of values rather than values, inheritance- inheritance -- all this classically suggests a production system architecture is needed. That is, the encoding of inheritance categories as production rules. There are two conflict resolution issues for the control structure of such an architecture: which rules to invoke, and how to resolve different answers from different rules.

Many different inference paths can be followed in making a statistical estimate, even not including all the possible rearrangements of a set expression involving intersections, unions, and complements. Since these can give different final answers, it's important to explore as many of these in parallel as possible, unlike most production systems where a single "best" alternative is desired. But some limits to parallelism have to be set for complicated queries, and we are currently investigating "weakest-first" inference. (Arithmetic must be generalized for operations on intervals.)

Combining results from different inference paths is straightforward for numeric statistics. Intersect the ranges to get a cumulative range. Get the cumulative estimate by assuming independence for all estimates, combining as if their errors were characterized by normal distributions via the classical statistical formulas; and the cumulative standard deviation follows directly. Even with nonindependence in the latter calculations the estimate should not be off much, and the standard deviation for the two-path case is never more than 70% ($2^{1/2}$) of what it should be.

2.6 An application

We are implementing a program that uses these ideas to answer statistical questions for a large database, discussed in more detail in the first subpaper in this memo. It uses several hundred rules from a variety of sources: mathematical definitions, extreme-value analysis of definitions, statistical theorems, exploratory data analysis, database dependency theory, statistical database inference security research, psychology of conceptual classes, and general principles of information systems. As with many other "expert systems" in artificial intelligence, there is more fundamental mathematical theory -- in this case, nonlinear optimization and cross-entropy minimization [18] -- that underlies many of the rules, but is too intractable for all but the simplest cases to be of much use.

3. Diophantine compromise of a statistical database

A statistical database is said to be compromisable if individual data items can be inferred from queryable values of statistical aggregates (mean, maximum, count, etc.) ([10], ch. 13). It has not apparently been recognized that solution of Diophantine equations is a powerful compromise method.

We formulate the problem in Diophantine terms in part 1. In part 2 we review the standard methods for solving such equations. In parts 3 and 4 we discuss how to improve equation-solving performance by getting additional equations and additional constraints. In part 5 we discuss countermeasures, in part 6 another application of these ideas, and in part 7 some conclusions.

3.1 Problem definition

There are two cases. The more important one is when exact statistics queryable on some class (set) of items (objects) include:

1. the class size

2. the mean with respect to the values of some attribute

3. the set of all possible values for that attribute, and (practically speaking) the number of such values is small compared to the size of the set

There is also a dual problem, where what are known are:

1. the class size

2. the mean with respect to the values of some attribute

3. the number of occurrences of each possible value for that attribute, and (practically speaking) the number of such occurrences is small compared to the size of the set.

In either case we can write the simultaneous linear Diophantine equations

$$n_1 + n_2 + n_3 + ... = n$$

$$(n_1v_1 + n_2v_2 + n_3v_3 + ...) / D = \mu n / D$$

where n is the set size, μ the mean, the v_i's the possible values, and the n_i's their occurrence counts. D is the greatest common divisor of the coefficients (the v_i's in the main case, the n_i's in the dual case); it must be introduced to assure integral coefficients. So for instance if the v_i's are 3.14, 6.5, and 1.52, and μ is 531, then D is .02 and the equation can be written $107v_1 + 325v_2 + 76v_3 = 26550$.

In the main case the n_i's are unknown; in the dual case, the v_i's. In this paper we shall be concerned, unless otherwise stated, with the main case instead of the dual. Usually values are more fixed than counts of sets that have those values, and often they can be looked up in books and found in other ways than querying the database[1].

Why is solution of this equation set compromising? Perhaps not so much for the dual problem, where the values found are often not very important. But in the main problem we may be able to infer that certain values are relatively rare. A value with a count of 1 is a data value revelation. A value with a count of 0 is a "negative compromise" that tightens other inferences. Other small counts can lead to small-set inferences [8].

3.2 Solving the equations

Linear Diophantine equations (unlike most other Diophantine equations [13]) can be solved algorithmically. It can be shown that solutions in the unconstrained case (no bounds on any variables) form a set of evenly spaced lattice points in hyperspace[2]. Thus they can be completely characterized by a single point x_0 and a set of basis vectors, multiples of which represent offsets from x_0 in each dimension. Knuth ([11], p.326-327) provides an elegant algorithm for obtaining this basis, based on the Euclidean greatest-common-divisor algorithm.

This done, constraints on the solution such as bounds on variables can be imposed in a second phase. In the usual (not dual) problem, the unknowns are counts which are necessarily nonnegative and individually no larger than n, their known total sum. Call vector w the set of integer multiples of the basis vectors needed to be added to some solution x_0 to reach some arbitrary solution x. Then by the Simplex method we can find upper and lower bounds on each variable (component) of w by working with the set of linear equations corresponding to the basis vectors, optimizing a function whose value is the value of each single variable in turn. Then if we are asked to generate all solutions to the original Diophantine equation satisfying constraints, we know we need only search the finite number of points lying in a bounded hyperrectangular region. We can use search algorithms from artificial intelligence [14] to explore the combinations.

[1] The two cases can be cascaded together, allowing inference of the *mean* of an attribute. Suppose you know the values of one attribute, the number of items having specific values for a second attribute, and you know that the corresponding attribute values are in one-to-one correspondence, as will be discussed in 3.2. You can infer the mean (as well as median, mode, standard deviation, etc.) of the second attribute.

[2] Observe that if x_1 and x_2 are two solutions to a set of linear Diophantine equations, then $x_1 + k(x_2 - x_1)$, where k is an integer, is also a solution.

Phase 1 is based on the Euclidean algorithm, and hence extending the analysis of [11] p. 323, should be O(m $\log_2 k$), where m is the number of equations, and k is the largest absolute value of a coefficient in any equation. This will not be bad for most practical cases, and phase 2 will more likely dominate. Phase 2 involves individual checking of solution combinations, and number of such combinations can be large. Since their number depends on complex number-theoretic properties of the integer coefficients of the original equations, it is hard to know in advance how many there will be. A much larger class may turn out to have a much *smaller* set of solutions to its Diophantine equation than a smaller class.

3.3 Additional evidence: multiple equations

There are a number of ways that additional equations can be generated for the same variables, thus providing even smaller solution sets and faster solution times.

3.3.1 Equations from moments

If for some set we know the standard deviation in addition, i.e. we know:

1. the set size
2. the set mean with respect to some attribute
3. the set standard deviation with respect to the same attribute
4. all possible values for that attribute

we can write an additional linear Diophantine equation on the same variables:

$$[n_1(u_1-\mu)^2 + n_2(u_2-\mu)^2 + n_3(u_3-\mu)^2 + \ldots] / (D_u)^2 = \sigma^2 / (D_u)^2$$

to solve simultaneously with our original two. And if we can also get higher-order moments on that attribute, they generate additional linear equations as well[3].

3.3.2 Equations from linked attributes

Another way to get an additional equation is when we know that the values for two numeric attributes u and v of some object are in one-to-one correspondence -- that is, when a particular value for u logically determines the value for v, and vice versa. (This does not necessarily mean a functional dependency in both directions, as the term is used in database research [23]; the relationship may be

[3] Note these are all linear equations for the main (unknown-counts) problem, but polynomials for the dual problem.

an "extensional functional dependency", a one-to-one mapping in only a particular database state.) Then if we know for some class:

1. the set size
2. the mean with respect to attribute u
3. the mean with respect to attribute v
4. all possible values for attribute u
5. all possible values for attribute v
6. the single fixed pairings of u values with v values

we can write three Diophantine equations to solve simultaneously:

$$n_1 + n_2 + n_3 + ... = n$$

$$(n_1u_1 + n_2u_2 + n_3u_3 + ...) / D_u = \mu_u n / D_u$$

$$(n_1v_1 + n_2v_2 + n_3v_3 + ...) / D_v = \mu_v n / D_v$$

If the u values are statistically independent of their correlated v values, the selectivities will tend to multiply in solving the combined equation. That is, the number of points satisfying the constraints and both of the two last equations will be the product of the number satisfying the constraints and each of the last two equations separately, divided by the number of points satisfying only the constraints.

As an example, suppose we know there are 58 ships in the British fleet in the South Atlantic. Suppose further that we know there are only three kinds of ships in that fleet, with lengths 440, 210, and 80 feet, and beams 52, 27, and 20 feet respectively. Suppose the mean length in the fleet is 190, and the mean beam 26. Then the equations are:

$$n_1 + n_2 + n_3 + ... = 58$$

$$(440n_1 + 210n_2 + 80n_3) / 10 = 190 * 58 / 10$$

$$(52n_1 + 27n_2 + 20n_3) / 1 = 26 * 58 / 1$$

3.3.3 Equations from one-way maps

The relationship between attributes u and v need not be one-to-one to use this approach. A functional mapping in either direction is sufficient, provided as before that we know which values map onto which other values. We need twice as many variables as previously, however: a set for u and a set for v. Assuming the mapping is from u onto v, we then write an additional set of equations equating the count for each variable in v with the sum of the counts variables of u that correspond.

So if variables 1 and 2 of u are the only ones that map onto variable 6 of v, we write $n_{v6} = n_{u1} + n_{u2}$. We then solve simultaneously the Diophantine set of the u equation, the v equation, the sum-of-counts equation, and these u-v interrelationship equations.

(The exact mapping of values is often "domain knowledge" that can be looked up in books or inferred by common sense. If not, something still can be done. Solve for the u and v count variables separately in phase 1. Then in phase 2, the generation of solutions, throw out any u solution which can't be summed in some way to get at least one v solution. But this is slow.)

3.3.4 Equations from nonnumeric attributes

Even nonnumeric attributes can be used to calculate means if type checking is not enforced on a database. For instance, computing the mean of a character-string field may involve converting the first two characters into a 16-bit integer. This gives a one-way functional mapping which can be exploited with the above ideas if the nonnumeric attribute itself is the object of a functional mapping from the target attribute.

3.3.5 Equations from virtual attributes

As if the above methods weren't enough, there are even more powerful ways of generating equations. Attributes need not be stored directly, but can be derived by numerical operations on data like logarithms, exponentials, square roots, squares, reciprocals, absolute values; and for pairs of attributes, operations like sums, differences, products, quotients, etc. So if a database system permits such operations before calculating means, this provides many usually independent equations. And even more if operations can be cascaded: you could take the mean of the sum of the logs, the mean of the log of the sums, as well as the mean of the sum, the mean of the logs, etc.

3.3.6 Equations from database updates

Inserts, deletes, and updates involving single items of classes followed by recalculation of the mean do not provide any new equations. For instance, inserting an item with value b for some attribute into a class of size m with mean μ for that attribute always gives a new mean of $(n\mu + b)/(n + 1)$. But if subclasses of indefinite size are inserted, deleted, or updated things are different; for example, the number of items with a certain value in a class is the number of items in the class minus the number of items in the set formed by deleting all items with that value from the class. Relational joins ([23], ch. 4) can disguise such conditions when they are permitted the user. For instance, to find how many items in a class have a given attribute, join its relation to another relation where that attribute is a key.

Let the second relation have another field f which is numeric. Then by computing the mean μ_0 of the field f in the join, changing the entry in f for the target from v_0 to v_1, and finding the new mean μ_1, we can determine the target value as

$$n(\mu_1 - \mu_0)/(v_1 - v_0)$$

where n is the size of the joined relation (and also the size of the original first relation). Note this is a different sort of compromise-by-update than usually discussed [3]; here known updates compromise unknown a priori circumstances, not the other way around.

As an example, suppose we want to infer the salary distribution of a set of 10,000 employees. Suppose we know a set of values certain to include all salaries. For instance, it might be all multiples of $1000 from $5000 to $100,000, if we know it is company policy to only pay in such multiples. We then create a new relation whose key is this salary value and whose dependent part is a single attribute whose value is always 0 except for the first entry, for which it is 1. We then compute the mean of this attribute for the join of the main and new relation -- suppose it is .0074. Then there must be

$$10000 * (.0074 - 0) / (1 - 0) = 74$$

employees with that first salary value.

3.3.7 Equations from multiple factorizations

Even when sophisticated kinds of processing such as virtual attributes and updates are not permitted on a statistical database, and no functional mappings appear between attributes, additional evidence may be obtained another way. Contingency tables (as the term is used in statistics) may be set up, and solved for the values in cells from the marginals that represent sums of rows and columns (and levels, etc. if the tables are more than 2-dimensional). This requires a database in which an attribute whose means can be queried has itself a class partitioning whose means for other numeric attributes can be queried. Number theoretic investigations of magic squares are relevant here.

As an example, suppose we have a military ships database with attributes ship nationality, ship class, and length. Assume a ship's nationality and class uniquely determine its length (i.e., there is a functional dependency from them to length). Our unknowns are the number of ships of each nationality-class pair. For each ship nationality we can write a Diophantine equation using the number of ships of that nationality and their mean length. For each ship class we can write a Diophantine equation using the number of ships of each class and their mean length. Thus we have two sets of equations involving the same unknowns in different ways.

3.4 Additional evidence: multiple constraints

Additional equations simplify phase 1 of the Diophantine solution algorithm. Additional evidence may also help in phase 2 in the form of additional constraints to rule out solutions as they are generated.

As an example, consider information about the frequency distribution of values. If a statistical database allows you to compute the mode of a class, it should also give the frequency of that mode in order to quantify its significance, analogously to a standard deviation quantifying the significance of a mean. A mode frequency not only lets you eliminate a variable from your Diophantine equations (the variable for the mode value's count), but puts an upper bound on all other count variables. If the system won't tell you the mode frequency, bounds on it and other frequencies may be obtainable from other counts queryable on the database -- so for instance the number of destroyers with "Russia" as the value of their nationality attribute can't be any more than the total number of Russian ships. Or you may be able to solve a simpler Diophantine problem on a superset of the target set (remember, simplicity is sensitive to low-order bits of numbers involved) and use these counts as upper bounds on the corresponding counts for the target set. Even when these methods won't work, frequencies can be estimated with "reasonable guesses" by experts familiar with the domain of the database, and also by general rules-of-thumb like Zipf's "Law" that the frequency of the kth most common item is proportional to 1/k. There are also absolute, mathematically provable bounds that can apply. For instance, the mode frequency of a class can't be any less than the number of tuples in the class divided by the number of distinct values.

Order statistics like median and quartiles are another source of constraints. For instance, knowing the size and median of a class tells you the exact number of items above and below that median value. Maxima and minima can also give constraints. If you know that the distribution of values for some attribute is even to some tolerance, and you know the density of values, then you can put bounds on the size of a set given its maximum and its minimum. If you combine this with the fact that the maximum of the intersection of a bunch of sets is the minimum of the maxima, and the minimum of the intersection is the maximum of the minima, you can sometimes infer narrow ranges of values for the items of an intersection, and hence tight upper bounds on the sizes of intersection sets.

Counts (and bounds on them) can be inferred by a large variety of indirect methods besides these. Ongoing work ([15], [16]) of which this research is a small part is directed towards constructing a rule-based system to make such estimates, and estimates of other statistics, by a large collection of methods. Integrated inference systems of this sort can demonstrate synergistic effects from rule combinations. Note that such reasoning involves almost exclusively the high-order bits of numbers, whereas Diophantine analysis emphasizes number-theoretic considerations sensitive to low-order bits, and hence selectivities of non-Diophantine and Diophantine restrictions will tend to multiply. Thus both together can be powerful.

3.5 Countermeasures

As we have seen, Diophantine inferences can compromise a database from many different sources of information. Thus restrictions on the kinds of single queries a user of a database can ask may not be much of an impediment to compromise. Furthermore, protection by restricting queries with extreme (very large or very small) counts and query pairs with excessive overlap between target classes (cf. [4]) are nearly useless. Diophantine compromise can work on classes of any size (though it tends to work better for smaller classes), and need only involve queries on one class (set). A third possible countermeasure, database partitioning, doesn't prohibit Diophantine analysis, just restrict the number of sets to which it can be applied.

So there is only one countermeasure left in the list of [4]: random perturbations of data and/or query answers [21]. (For our purposes, data perturbations amount to the same thing as answer perturbations.) Note the perturbations must be pseudo-random, as opposed to truly random, or the user could ask the same question many times and average the results to get an answer with much less variance. Note also the perturbations must be large enough so the user cannot enumerate a small set of possibilities for set sizes and means, and then solve each possibility separately, throwing away possibilities that yield no solutions consistent with the constraints. With many equations and/or constraints, the average yield, based on selectivities, of the a priori information may be significantly less than 1 record, hence there may be many possibilities for means and sizes that are inconsistent, while the true possibilities are guaranteed to give solutions.

Thus far we have assumed we know *exact* values of the total size of the set being analyzed, its mean for some attribute, and its possible values. But with large classes, rounding and/or truncation will occur in finding means (and even class sizes). Thus a kind of random perturbation may occur "for free" in many statistical databases when large sets are queried, and specifically adding perturbations may not be necessary for protection. But first, the calculation error may not be sufficient for protection. Second, truncation and rounding are deterministic operations, not random, and may be analyzable. For instance, the values may be known to be sorted before they are added, or there may be only one particular value with low-order bits that will be significantly rounded; in both these cases large chunks of possibilities can be eliminated, meaning much less protection from compromise. Clever users can create complex compromise methods based on detailed scenarios for rounding and/or truncation.

3.6 Other uses of Diophantine inference

To be fair, these compromise methods can be used for good as well as evil. As we discuss in [15] and [16], even simple statistical calculations like counts and means can take enormous amounts of time in a very large database. But as computer technology progresses, more and more data is being stored. Many users of such data would be gladly willing to accept approximate answers to some statistical queries on these huge databases, answers based on inferences including our Diophantine methods, in return for greatly increased access speed and/or greatly decreased storage requirements -- particularly in the early stages of exploratory data analysis [22].

3.7 Conclusions

Diophantine compromise represents a serious new threat to the security of statistical databases. It can be applied to find exact sizes of subclasses of a given class of items in a database, given the size, mean, and set of values for that class -- things which are not intrinsically suspicious. While solving a single such Diophantine equation may generate many solutions, too many to be very compromising, there are a wide variety of ways to cut down the number of possibilities by getting both additional simultaneous equations and constraints. One major countermeasure is possible, random perturbations of answers supplied to the user, but it has disadvantages, in particular its degradation of answer quality and being subject to clever exploitation if not applied carefully. Clearly this topic deserves detailed further investigation.

References

[1] R. J. Brachman and D. J. Israel.
KL-ONE Overview and Philosophy.
In W. A. Woods (editor), *Research in Knowledge Represenation for Natural Language Understanding: Report No. 4785*, pages 5-26. Bolt Beranek and Newman, 1981.

[2] Jaime G. Carbonell.
Default Reasoning and Inheritance Mechanisms on Type Hierarchies.
In *Proceedings*, pages 107-109. Workshop on Data Abstraction, Databases, and Conceptual Modelling, Pingree Park CO, June, 1980.

[3] F. Y. Chin and G. Ozsoyoglu.
Security in Partioned Dynamic Statistical Databases.
In *Conference Proceedings*, pages 594-601. IEEE COMPSAC, June, 1979.

[4] Francis Y. Chin and Gultekin Ozsoyoglu.
Statistical Database Design.
ACM Transactions on Database Systems 6(1):113-139, March, 1981.

[5] Allan Collins.
Fragments of a Theory of Human Plausible Reasoning.
In *Proceedings*, pages 194-201. Second Conference on Theoretical Issues in Natural Language Processing, Urbana IL, July, 1978.

[6] Allan Collins, Eleanor H. Warnock, Nelleke Aiello, and Mark L. Miller.
Reasoning From Incomplete Knowledge.
Academic Press, New York, 1975, pages 383-415.

[7] Randall Davis and Jonathan King.
An Overview of Production Systems.
In E. W. Elcock and D. Michie (editor), *Machine Intelligence 8*, pages 300-334. Wiley, New York, 1976.

[8] D. E. Denning, P. J. Denning, and M. D. Schwartz.
The Tracker: a Threat to Statistical Database Security.
ACM Transactions on Database Systems 4(1):76-96, March, 1979.

[9] Richard O. Duda, Peter E. Hart, Nils J. Nilsson, Rene Reboh, Jonathan Slocum, and Georgia L. Sutherland.
Development of a Computer-based Consultant for Mineral Exploration.
Annual Report Projects 5821 and 6415, SRI International Artificial Intelligence Center, Menlo Park, CA, October, 1977.

[10] Eduardo B. Fernandez, Rita C. Summers, and Christopher Wood.
Database Security and Integrity.
Addison-Wesley, Reading MA, 1981.

[11] Donald E. Knuth.
The Art of Computer Programming, volume 2: Seminumerical Algorithms.
Addison-Wesley, Reading MA, 1981.

[12] D. B. Lenat, F. Hayes-Roth, and P. Klahr.
Cognitive Economy.
Working Paper HPP-79-15, Stanford University Heuristic Programming Project, June, 1979.

[13] L. J. Mordell.
Diophantine Equations.
Academic Press, New York, 1969.

[14] Nils Nilsson.
Principles of Artificial Intelligence.
Tioga, Palo Alto, 1980.

[15] Neil C. Rowe.
Rule-Based Statistical Calculations on a Database Abstract.
In *Proceedings*, pages 163-176. First LBL Workshop on Statistical Database Management, Menlo Park CA, December, 1981.

[16] Neil C. Rowe.
Inheritance of Statistical Properties.
In *Proceedings of the National Conference*, pages 221-224. American Association for Artificial Intelligence, Pittsburgh PA, August, 1982.

[17] Stuart C. Shapiro.
Path-Based and Node-Based Inference in Semantic Networks.
In *Proceedings*, pages 219-225. Second Conference on Theoretical Issues in Natural Language Processing, Urbana IL, July, 1978.

[18] John E. Shore and Rodney W. Johnson.
Properties of Cross-Entropy Minimization.
IEEE Transactions on Information Theory IT-27(4):472-482, July, 1981.

[19] Edward Shortliffe.
MYCIN: A Rule-Based Computer Program for Advising Physicians Regarding Antimicrobial Therapy Selection.
Technical Report STAN-CS-74-465, Stanford University, Stanford CA, October, 1974.

[20] John F. Sowa.
A Conceptual Schema for Knowledge-Based Systems.
In *Proceedings*, pages 193-195. Workshop on Data Abstraction, Databases, and Conceptual Modelling, Pingree Park CO, June, 1980.

[21] J. F. Traub, H. Wozniakowski, and Y. Yemini.
Statistical Security of a Statistical Data Base.
Technical Report, Columbia University Computer Science Department, September, 1981.

[22] John W. Tukey.
Exploratory Data Analysis.
Addison-Wesley, Reading, Mass., 1977.

[23] Jeffrey D. Ullman.
Principles of Database Systems.
Computer Science Press, Potomac MD, 1980.

[24] Gio Wiederhold.
Database Design.
McGraw-Hill, New York, 1977.

[25] Lofti Zadeh.
A Theory of Approximate Reasoning.
In E. W. Elcock and D. Michie (editor), *Machine Intelligence 9*, pages 149-196. Wiley, New York, 1979.